Universitatea din București
Facultatea de Matematică-Informatică
Proiect Structuri de Date Avansate

Bază de date pentru evidența publicațiilor și a abonamentelor la o firmă de distribuție a presei

Student Gafencu Mihai-Costin

Ianuarie, 2018

Contents

1 Introducere

O firmă de distribuție a presei are nevoie pentru a-și desfășura activitatea de o organizare coerentă și colerată a datelor cu care sunt nevoiți să lucreze. De aceea, baza de date creată se va ocupa de gestiunea informațiilor primite de la redacții, dar si de la abonați. Firma prelucrează datele abonaților (nume, locație, publicațiile pe care trebuie sa-le primească, numărul de telefon) pentru a îndeplini cerințele redacțiilor; astfel trebuie să existe o organizare a locațiilor de expediere, dar și a informațiilor cu privire la publicati. Angajați firmei se ocupă de împachetarea și livrarea comenzilor la adresele cerute, dar și de ridicarea publicatiilor de la diversele redacți, care vor fi duse pentru împachetare în depozitul cel mai apropiat de abonat.

Baza de date va conține informații despre angajați firmei, abonați săi, publicatiile, locatiile clientilor, dar si locatiile redactiilor, de unde trebuiesc ridicate publicațiile. Va fi suficient doar să reținem datele angajatilor care se ocupă cu distributia editorialelor, pentru a cunoaste cine dintre acestia opereaza intr-un anumit oras (specificam ca un angajat nu poate lucra in mai mult de un oras); datele despre restul angajațiilor reprezentând doar informații suplimentare. De asemenea, clientul nu trebuie să primeasca duplicate, numai dacă dorește acest lucru, iar numărul cerintelor trebuie să corespundă cu numărul de publicati ridicate de la redacții. Referitor la platile abonamentelor, firma de distributie nu se ocupa cu acest fapt, aceasta reprezentand doar un intermediar între abonat și redacție.

Abonați pot fi de doua tipuri: persoane fizice sau juridice. Chiar dacă acestea au anumite atribute similare, trebuie facută o diferență strictă între acestea două, fiind spre beneficiul redacțiilor. Un individ se consideră abonat atâta timp cât ii corespunde o publicație, abonamentul fiind descris de data de început si de sfârșit al abonarii. O publicație trebuie să corespunda unei singure redacții, iar asemanător ca in cazul abonamentului, publicația este descrisa de data apariție și numărul ediției.

2 Definirea entităților

Prima entitate a bazei va fi denumită REDACTII, care descrie datele redactiilor (numele, locație, director și publcațiile care îi corespund). În plus, în tabel trebuie reținută o modalitate de contact pentru a putea remedia diversele erori ce pot aparea in distribuție. Cheia primara a acestui tabel va fi reprezentat de id-ul numeric specific fiecărei înregistrări, RedId.

Entitatea urmatoare a bazei va fi PUBLICATII, care va conține datele diverselor publicați, dar și informați referitoare la prețul, categoria în care se încadrează. Prin datele publicațiilor întelegem numele, data apariție, numărul ediției, redactia de care apartine, redactorul sef al acesteia. Pentru a diferența informațiile din tabel vom utiliza o coloana cu valoarea numerică unică, denumită PubId.

Entitatea ABONATI va reține informațiile despre abonați, cum ar fi nume, prenume, adresa si modalitatile de contact. Cum am specificat anterior, pot exista doua tipuri de abonati (persoane fizice sau jurdice); fiind necesar să diferențiem cele doua categorii. Pentru distribuție este necesar cunoașterea adresei de livrare exacte a persoanelor. Fiecare abonat este identificat în mod unic prin coloana cu valoarea numerică unică AbnId, care reprezinta cheia primară.

Ultima entitate din bază poartă denumirea ANGAJATI, care va reține numele fiecărui angajat al firmei, orașul în care acesta operează. Este necesar cunoașterea angajaților care operează într-un anumit oras, pentru a putea preveni eventualele erorii ce pot apărea în distribuția abonamentelor. Fiecare angajat este identificat în mod unic printr-un id numeric, denumit AngId, acesta fiind cheia primă a tabelului.

De observat faptul că în tabelul ABONATI sunt incluse două subentități, care reprezintă tipul de abonament. Cele două subentități corespunzătoare sunt: PERSFIZ (peroanele fizice) și PERSJUR (persoanle juridice). Cele doua au atribute similare, dar se diferentiază printr-o serie de atribute caracteristice fiecăruia (spre exemplu o firma trebuie să aibă un director sau manager).

3 Relații

Legături se stabilesc între următoarele tabele: REDACTII-PUBLICATII, PUBLICATII-ABONATI.

Relația REDACTII-PUBLICATII se caracterizeaza prin faptul că unei publicați îi corespunde o singură redacție, cea care a scris, eventual tipărit editorialul; iar o redacție poate să editeze mai multe feluri de publicatii, nefiind restrictionată la o anumită categorie sau tip de editorial. Specificăm că, în baza de date nu luăm în considerare publicațiile care reprezintă o colecție de articole, aparținând mai multor redacții, întrucat astfel de editoriale (anume almanahuri) sunt rarisime și pot fi ditribuite, eventual, ca bonusuri. Astfel, relația "redacția

editează publicația" sau "publicația tipărită de redacția" are cardinalitatea maximă *one-many* $(1 : n)$. Iar, o redacție nu trebuie să tipărească neapărat vreo publicație; relația redactia editeaza publicatia având astfel cardinalitatea minimă *1-zero* $(1 : 0)$; deoarece un editorial este neapărat publicată de către o redacție.

Relația PUBLICATII-ABONATI se descrie prin idea că o publicație poate să aibă mai multi abonați; iar o persoana poate să dețină mai multe abonamente. Prin urmare, relatia "persoana abonată la publicația" are cardinalitatea maximă *many-many* $(m : n)$. Pe când cardinalitatea minimă a relației anterioare este *zero-one* $(0 : 1)$, întrucât o publicație nu trebuie să aiba neapărat un abonat, dar cazul un abonat fară publicație este imposibil (din felul în care am definit anterior ce prezintă un abonat în tabelul ABONATI).

Mai există relațiile de tipul ISA, stabilite între superentitatea ABONATI cu subentitățile PERSFIZ si PERSJUR. Conform teorie, aceasta relatie are cardinalitatea maxima *one-one* $(1-1)$, respectiv cardinalitatea minima *one-zero* $(1-0)$. Acest fapt se justifică prin idea că o persoana corespunde entități la care e angajată, iar firma ce realizează angajarile poate să aibă 0 sau cel mult un angajat pe un anumit post.

4 Descrierea atributelor

Prima entitate, REDACTII, este caracterizată de următoarele atribute:

- RedId -> menționată anterior în paragraful ce descrie entiatea, este de tip numeric (*integer*) , este unică și are rol de cheie primară.
- Nume -> sugerează denumirea redacție. Acesta nu trebuie să aibă valoare nul, fiecare redacție având o denumire. Este de tip șir de caracatere (*char*).
- Adresa -> descrie locația exactă a redacției. Are tipul șir de caractere (*char*) și nu trebuie să memoreze valoare nul. Prin locație exactă întelegem că pe lângă stradă, numărul, codul poștal, se mai specifică orașul si județul; ultimele două informați find utile pentru distribuție.
- Telefon -> menține numărul de telefon al redacției. Este de tip șir de caractere (*char*) și nu poate să fie nul, întrucât trebuie să avem acces la o modatalitate de contact a redacției. În ceea ce privește dimensiunea șirului, trebuie să fie de lungime 10, pentru că împreuna cu prefixul un număr de telefon de mobil/fix are exact 10 cifre, și începe cu cifra 0. Un contraexemplu ar fi orice șir de forma '3456013421'.
- Email -> are valorile de tip șir de caractere (*char*) și reține email-ul redacției. Un exemplu corect de email trebuie sa conțină extesia '@', pe când un email fără '@' poate fi considerat invalid.
- RedactorSef -> reprezintă numele persoanei care conduce efectiv redacția. Are valorile de tip șir de caractere (*char*).

Tabelul PUBLICATII, este descris de urmatoreale atribute:

- PubId -> menționat la fel ca RedId în paragraful ce descrie entitățile, este cheia primară și reține valori numerice unice (*integer*).
- Nume -> avand tipul sir de caractere (*char*), reține denumirea publicației. Acesta nu trebuie să abia o valoare nulă, fiecare editorial trebuind să aibă o denumire. Numele trebuie sa fie cât mai sugestiv, având scopul de a atrage atenția și interesul cititorilor. Acesta nu trebuie să includă informați inutile, să aibă o limită prestatibilită de caractere acceptate, întrucât titlul editorialul trebuie să fie ușor de reținut și de distins. Un nume corespunzător ar putea fi: "Cotidianul de Seara" sau "Dilema Veche", "Dilema Noua". Altfel, un titlu nesugestiv este: "Prezentarea situației sociale din trecut și prezent

din perspectiva unui deontolog"; pe când titlul mai sugestiv ar fi: "Concepțiile unui Deontolog".

- Aparitie -> sugerează data de aparție a editorialului, reține valori de tip date (*date*). Data trebuie să fie reprezentată prin zi, luna și an. O valoare acceptabilă are forma, spre exemplu: '11.12.2017'. Valoarea s-a nu trebuie sa fie nulă, pentru a putea realiza ulterior o staticata asupra publicatiilor aparute pe o anume data.

- NrRed -> reprezintă o valorea numerica unică nenulă (*integer*) ce realizeaza legătura cu tabelul REDACȚII. Valorile sale sunt corespunzatoare cu valorile atributului RedId; astfel, NrRed care are rol de cheie externa.

- NrEditiei -> menține numărul de aparației al editorialului. Reprezintă valori de tip numerici (*integer*). Acesta sugerează numărul de apariție într-un anumit an al editorialului. Spre exemplu, valoare 3 ar sugera că publicația este cea de-a treia editie dintr-un an.

- Categorie -> fiecare publicație poate fi inclusă într-o anumită categorie. Are valoarea de tip șir de caractere (*char*). O categorie poate fi, de exemplu, sport sau politică. Denumirea trebuie să fie cât mai sugetivă, specificând în mod cât mai clar numele categoriei din care face editorialul. Este posibil ca o publicație să includă mai multe subcategorii, atunci se va memora categoria 'generalist'; deoarece o revistă sau ziar include o varietate de subiecte (articole din diverse domenii). Nu se acceptă categorii de tipul 'politică-sport', deoarece acest tip de editorial poate fi inclus în categoria 'generalist'.

- Ultimul atribut necesar în descrierea tabelului este Pret (reprezintă prețul unei publicați). Este de tip float și nu trebuie să fie nulă. Valorile atributlui trebuie să fie pozitive.

Următorul tabel, ABONATI, se descrie prin atributele:

- AbnId -> apare anterior în paragraful ce descrie entitățile, este cheia primară și reține valori numerice unice (*integer*).

- TipPers -> tipul este șir de caractere (*char*), nu trebuie să fie nul și reprezintă tipul abonamentului. Poate avea doar două valori, anume 'persfiz' sau 'persjur', semnificând persoana fizică, respectiv juridică.

- Nume -> sugerează numele persoanei sau a firmei care a realizat abonamentul. Este de tip șir de caractere (*char*) și nu trebuie sa fie nulă.

- tipjur -> reprezintă tipul juridic al firmei. Este de tip șir de caractere (*char*) și nu trebuie sa fie nul.

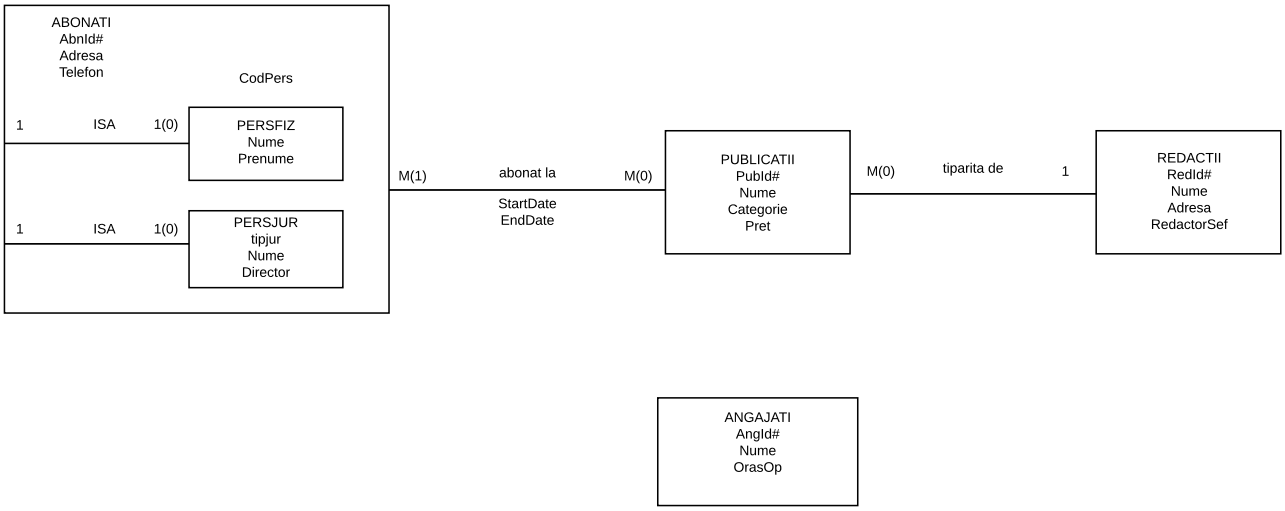
- Director -> reprezintă numele persoanei care conduce firma care a realizat abonamentul. Are valorile de tip șir de caractere (*char*).
- Telefon -> menține numărul de telefon al abonaților. Este de tip șir de caractere (*char*) și nu poate să fie nul. Este asemănator cu atribul Telefon asociat redacțiilor.
- Email -> are valorile de tip șir de caractere (*char*) și reține email-ul redacției. Are aceleași caracteristici cu atributul Email al entități REDACȚII.
- Adresa -> descrie locația exactă a abonaților. Are tipul șir de caractere (*char*) și nu trebuie să memoreze valoare nul. Prin locație exactă întelegem că pe lângă stradă, numărul, codul poștal, se mai specifică orașul si județul; ultimele două informați find utile pentru distribuție.
- PubId -> realizează legătura cu tabelul PUBLICAȚII și reprezintă publicațiile la care s-a abonat o persoana. Este de tip integer unic și nu poate fi nul.
- NrPub -> valoare numerică unică (tipul integer) și nu poate fi nul, iar valoarea sa trebuie să fie strict pozitivă. Acest atribut dă numărul de duplicate pe care le dorește abonatul.
- StratDate și EndDate -> semnifică data de început, respectiv de sfârșit a abonamentului. Aceste atribute descriu efectiv relația dintre ABONAȚI și PUBLICAȚII. Sunt de tip date, iar EndDate este nepărat mai mare decât StartEnd. Data nu trebuie să fie nulă și să fie reprezentată prin zi, luna și an. O valoare acceptabilă are forma, spre exemplu: '11.12.2016'.

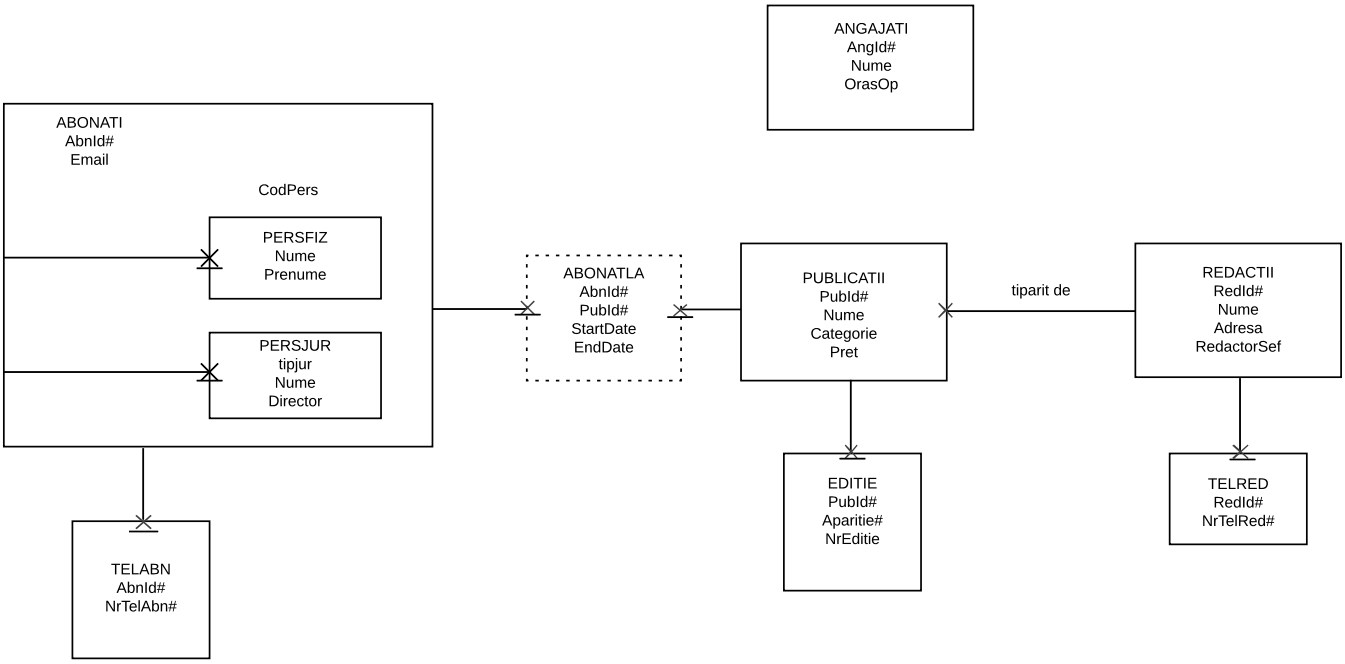
  Ultima entitate, ANGJATI, are următoarele atribute:

- AngId -> cheia primară, este de tip integer.
- Nume -> este de tip șir de caractere (*char*), nu trebuie să fie nulă. Reține numele angajatului.
- OrasOp -> prezintă orașul în care opereaza angajatul. Are valorile de tipul șir de caracter (*char*) și nu poate avea valoarea nul, întrucât este necesar reținerea orașului în care operează un angajat.

5 Diagrame. Scheme relaționale

Diagrama E/R și diagrama coneptuală:


ABONATI
AbnId#
Adresa
Telefon
CodPers
1
ISA
1(0)
PERSFIZ
Nume
Prenume
1
ISA
1(0)
PERSJUR
tipjur
Nume
Director
M(1)
abonat la
M(0)
StartDate
EndDate
PUBLICATII
PubId#
Nume
Categorie
Pret
M(0)
tiparita de
1
REDACTII
RedId#
Nume
Adresa
RedactorSef
ANGAJATI
AngId#
Nume
OrasOp


ANGAJATI
AngId#
Nume
OrasOp
ABONATI
AbnId#
Email
CodPers
PERSFIZ
Nume
Prenume
PERSJUR
tipjur
Nume
Director
ABONATLA
AbnId#
PubId#
StartDate
EndDate
PUBLICATII
PubId#
Nume
Categorie
Pret
tiparit de
REDACTII
RedId#
Nume
Adresa
RedactorSef
TELABN
AbnId#
NrTelAbn#
EDITIE
PubId#
Aparitie#
NrEditie
TELRED
RedId#
NrTelRed#

Schemele relaționale corespunzătoare sunt următoarele:

- REDACTII(RedId# , Nume, Oras, Judet, Adresa, RedactorSef);
- PUBLICATII(PubId# , Nume, Categorie, Pret, NrRed);
- ABONATI(AbnId# , Email, Adresa, Oras, Judet, CodPers);
- ABONATLA(AbnId# , PubId# , StartDate, EndDate, NrPub);
- ANGAJATI(AngId# , Nume, OrasOp, JudetOp);
- PERSIFZ(AbnId# , Nume, Prenume);
- PERSJUR(AbnId# , Nume, tipjur, Director);
- TELABN(AbnId# , NrTelAbn#);
- EDITIE(PubId# , Aparitie# , NrEditie);
- TELRED(RedId# , NrTelRed#);

Diagrama conceptuală a rezolvat pe lângă relațiile many-many, și posibilitatea ca abonați, redacțiile sau angajați să aibe mai multe numere de telefon.

6 Normalizare

În continuarea urmărim normalizarea bazei de date. Pornim cu prima schemă relațională, REDACTII. Se poate observa că toate atributele sunt atomice. Astfel, relația îndeplinește condiția primei forme normale, dar și al formei a doua normale (toate atributele care nu fac parte din cheia primară sunt depedente de întreaga cheie primară). De asemenea, se observă că atribule nu se află în condițiile formei a treia, deoarece, de exemplu, atributele Judet și Oras sunt în mod direct determinate de Adresă.

Din acest motiv vom crea un nou tabel denumit ADRESE, care va include adresele redacțiilor și ale abonaților. Tabelul va conține atributele:

- Adresa -> codul poștal al adresei, de tip integer, este cheia primără.

- Localitate -> de tip char, reține numele localități. Are valoare 'not null'.
- Judet -> de tip char, reprezintă județul în care se află localitatea; valoarea sa nefiind nulă.

Noile scheme relaționale vor fi:

- REDACTII(RedId# , Nume, Adresa, RedactorSef);
- ABONATI(AbnId# , Email, Adresa, CodPers);
- ADRESE(Adresa# , Localitate, Judet);

Totuși, în schemele anterioare se poate observa dependența dintre Localitate și Judet. Astfel, vom mai crea un tabel ce va conține județele și localitățile, din județul respectiv. Schemele relaționale devin:

- ADRESE(Adresa# , JudId);
- JUDETE(JudId# , Judet# , Localitate);

Astfel, prin schema relațională JUDETE, rezolvăm și dependeța dintre JudetOp și OrasOp din tabelul ANGAJATI.

ANGAJATI(AngId# , Nume, JudId);

Noua schemă relațională ANGAJATI îndeplinește condițiile FN1, FN2 și FN3, întrucât orice atribut non-prim nu este determinat funcțional de un al atribut non-prim.

Prin cele prezentate anterior, am încercat minizarea reduntanțelor ce pot apărea în tabel.

7 Anexe

```
create table REDACTII(RedId int, Nume varchar2(35) NOT NULL, Adresa
varchar2(50) NOT NULL, RedactorSef varchar2(50));

ALTER TABLE REDACTII ADD PRIMARY KEY(REDID);

create table PUBLICATII(PubId int, Nume varchar2(50) NOT NULL, Categorie
varchar2(20), Pret Number(8,2) NOT NULL, NrRed int NOT NULL);

ALTER TABLE PUBLICATII ADD PRIMARY KEY(PubId);

ALTER TABLE PUBLICATII ADD FOREIGN KEY(NrRed) REFERENCES REDACTII(RedId)
ON DELETE SET NULL;

create table ABONATI(AbnId int, Email varchar2(50), Adresa varchar2(50),
CodPers varchar2(5) NOT NULL);

ALTER TABLE ABONATI ADD PRIMARY KEY(AbnId);

create table PERSFIZ(AbnId int, Nume varchar2(25) NOT NULL, Prenume
varchar2(25));
create table PERSJUR(AbnId int, Nume varchar2(25) NOT NULL, tipjur
varchar2(10) NOT NULL, Director varchar2(20) NOT NULL);

ALTER TABLE PERSFIZ ADD PRIMARY KEY(AbnId);
ALTER TABLE PERSJUR ADD PRIMARY KEY(AbnId);
ALTER TABLE PERSFIZ ADD FOREIGN KEY(AbnId) REFERENCES ABONATI(AbnId) ON
DELETE SET NULL;
ALTER TABLE PERSJUR ADD FOREIGN KEY(AbnId) REFERENCES ABONATI(AbnId) ON
DELETE SET NULL;

create table ABONATILA(AbnId int, PubId int, StartDate date, EndDate
date, NrPub int NOT NULL);

ALTER TABLE ABONATILA ADD PRIMARY KEY(AbnId, PubId);
ALTER TABLE ABONATILA ADD FOREIGN KEY(AbnId) REFERENCES ABONATI(AbnId) ON
DELETE SET NULL;
ALTER TABLE ABONATILA ADD FOREIGN KEY(PubId) REFERENCES PUBLICATII(PubId)
ON DELETE SET NULL;
ALTER TABLE ABONATILA ADD CONSTRAINT Abdate CHECK(EndDate>StartDate);


create table EDITIE(PubId int, Aparitie date NOT NULL, NrEditie int);

ALTER TABLE EDITIE ADD PRIMARY KEY(PubId,Aparitie);
ALTER TABLE EDITIE ADD FOREIGN KEY(PubId) REFERENCES PUBLICATII(PubId) ON
DELETE SET NULL;


create table TELABN(AbnId int, NrTelAbn varchar2(15));

ALTER TABLE TELABN ADD PRIMARY KEY(AbnId,NrTelAbn);
ALTER TABLE TELABN ADD FOREIGN KEY(AbnId) REFERENCES ABONATI(AbnId) ON
DELETE SET NULL;

create table ADRESE(Adresa varchar(50), JudId int NOT NULL, Judet
varchar2(25));

ALTER TABLE ADRESE ADD PRIMARY KEY(Adresa);
```

```
create table JUDETE(JudId int, Judet varchar2(25), Localitate
varchar2(25) NOT NULL);

ALTER TABLE JUDETE ADD PRIMARY KEY(JudId,Judet);
ALTER TABLE ADRESE ADD FOREIGN KEY(JudId,Judet) REFERENCES
JUDETE(JudId,Judet) ON DELETE SET NULL;

create table TELRED(RedId int, NrTelRed varchar2(15));


ALTER TABLE TELRED ADD PRIMARY KEY(RedId, NrTelRed);
ALTER TABLE TELRED ADD FOREIGN KEY(RedId) REFERENCES REDACTII(RedId) ON
DELETE SET NULL;

create table ANGAJATI(AngId int, Nume varchar2(25) NOT NULL, Prenume
varchar2(25), JudId int, Judet varchar2(25));

ALTER TABLE ANGAJATI ADD PRIMARY KEY(AngId);
ALTER TABLE ANGAJATI ADD FOREIGN KEY(JudId,Judet) REFERENCES
JUDETE(JudId,Judet) ON DELETE SET NULL;
ALTER TABLE REDACTII ADD FOREIGN KEY(Adresa) REFERENCES ADRESE (Adresa)
ON DELETE SET NULL;
ALTER TABLE ABONATI ADD FOREIGN KEY(Adresa) REFERENCES ADRESE(Adresa) ON
DELETE SET NULL;
ALTER TABLE ABONATI ADD CONSTRAINT UniqueEmail Unique(Email);
```

```
--1

-- Sa se afiseze numele, pretul, redactia si data(luna/an) pentru
editoriale aparute in luna aprilie

select p.nume Nume, p.pret Pret, r.nume NumeRedactie,
to_char(e.aparitie,'mm/yyyy') DataAparitie from publicatii p, redactii r,
editie e
where (p.pubid=e.pubid) and (to_char(e.aparitie,'mm')='04') and
(p.nrred=r.redid);

--2

-- Afisati numele, pretul, redactia si anul apartiei pentru cele mai
ieftine publicatii, care au
-- aparut in luna aprilie.


select p.nume Nume, p.pret Pret, r.nume NumeRedactie,
to_char(e.aparitie,'yyyy') Anul from publicatii p, redactii r, editie e
where (e.pubid=p.pubid) and (to_char(e.aparitie,'mm')='04') and
(p.nrred=r.redid) and (p.pret=(select min(p1.pret) from publicatii p1,
editie e1 where (p1.pubid=e1.pubid) and
(to_char(e1.aparitie,'mm')='04')));

--3

-- Acelesa enunt ca la 2, numai ca folosim o procedura pentru a returna
datele cerute, procedura care primeste ca parametru
-- o luna( nu neaparat luna aprilie). Afisam toate publicatii, cu un
mesaj corespunzator daca pretul lor respecta sau nu conditia de minim a
pretului.
-- Tratam exceptiile.



create or replace procedure MinData (luna varchar2) as
minim number(8,2) :=0;
begin

for i in (select p.nume Nume, p.pret Pret, r.nume NumeRedactie,
to_char(e.aparitie,'yyyy') Anul from publicatii p, redactii r, editie e
where (e.pubid=p.pubid) and (to_char(e.aparitie,'mm')=luna) and
(p.nrred=r.redid)) loop

select min(p1.pret) into minim from publicatii p1,
editie e1 where (p1.pubid=e1.pubid) and (to_char(e1.aparitie,'mm')=luna);

if i.Pret=minim then
dbms_output.put(' Publicatia ' || i.Nume || ' are pretul ' || i.Pret ||
' apartine redactiei ' || i.NumeRedactie || ' si a aparut in luna aprilie
a anului ' || i.Anul);
else
 dbms_output.put('Publicati ' || i.Nume || ' nu indeplineste conditia de
minim!');
end if;
dbms_output.new_line;
end loop;

exception
```

```
when no_data_found then
raise_application_error(-20001,' Nu exista publicatii care au aparut in
luna specificata!');
when others then
raise_application_error(-20002, ' Alta eroare!');

end;
/

set serveroutput on;
accept mon prompt 'Introduceti luna:';
declare
l varchar2(4):='&mon';
begin
MinData(l);
end;
/
set serveroutput off;

drop procedure MinData;

--4
-- determinati numarul de publicatii din categoria 'Politic'
-- care au aparut in ultimii patru ani.

select count(*) Numar from publicatii p, editie e where
(p.pubid=e.pubid) and (months_between(sysdate,e.aparitie)<=48)
and (lower(p.categorie)='politic');

--5 enuntul anterior, exceptand categoria, dar considerand publicatiile
care
-- apartin de redactiile din judetul Alba.
-- in plus, mai calculam media preturilor ale acestora.

select count(*) Numar, avg(p.pret) Medie from publicatii p, editie e,
redactii r, adrese adr where
(p.pubid=e.pubid) and (months_between(sysdate,e.aparitie)<=48)
and (p.nrred=r.redid) and (r.adresa=adr.adresa)
and (lower(adr.judet)='alba');

--6 acelasi enunt ca la 4, dar pentru cele din categoria 'generalist',
-- care apartin de o redactie din judetul Alba.

create or replace procedure Numarare as
nr number:=0;
begin
select count(*) into nr  from publicatii p, editie e, redactii r, adrese
adr
where (p.pubid=e.pubid) and (months_between(sysdate,e.aparitie)<=48)
and (lower(p.categorie)='generalist') and (r.redid=p.nrred)
and (r.adresa=adr.adresa) and (lower(adr.judet)='alba');

if nr<>0 then
dbms_output.put_line(' In judetul Alba exista ' || nr || ' publicatii
generaliste care au aparut in ultimii patru ani!');
else dbms_output.put_line(' In judetul Alba nu exista publicatii
generaliste care au aparut in ultimii patru ani!');
end if;

end;
```

```
/

execute Numarare;

drop procedure Numarare;

--7
-- folosing o functie determinat numarul de publicatii dintr-o
-- categorie, care au aparut in ultimii doi ani. Tratam exceptiile.

create or replace function Numar (cat varchar)
return number as

nr number:=0;
begin

select count(*) into nr from publicatii p, editie e
where (p.pubid=e.pubid) and (months_between(sysdate,e.aparitie)<=24)
and (lower(p.categorie)=lower(cat));

if nr=0 then raise_application_error(-20012,' Categoria data nu are nicio
publicatie care a aparut in ultimii doi ani!');
end if;
return nr;

exception
when NO_DATA_FOUND then
raise_application_error(-20010,' Categoria specificata nu exista!');
when others then
raise_application_error(-20011,'Alta Eroare!');
end;
/

set serveroutput on;
accept cate prompt 'Introduceti categoria:';

declare
ct varchar(20):='&cate';
rez number;
begin
rez:=Numar(ct);
if(rez<>0) then
dbms_output.put_line(' In categoria ' || ct || ' sunt ' || rez || '
publicatii!');
end if;
end;
/
set serveroutput off;

drop function Numar;

--8

-- Folosind o functie determinati numarul de publicatii din
-- doua categorie, care au aparut in ultimii 3 ani.

create or replace type numbers is table of number;
/

create or replace function Numar1 (cat1 varchar2, cat2 varchar2)
```

```
return numbers as

numbs numbers:=numbers();
begin

numbs.extend();
select count(*) into numbs(1) from publicatii p, editie e
where (p.pubid=e.pubid) and (months_between(sysdate,e.aparitie)<=36)
and (lower(p.categorie)=lower(cat1));

numbs.extend();
select count(*) into numbs(2) from publicatii p, editie e
where (p.pubid=e.pubid) and (months_between(sysdate,e.aparitie)<=36)
and (lower(p.categorie)=lower(cat2));

if numbs(1)=0 or numbs(2)=0 then raise_application_error(-20021,' Una
dintre aplicatiile date nu are nicio publicatie aparuta in ultimii doi
ani!');
end if;
return numbs;

exception
when no_data_found then
raise_application_error(-20022,' Categoriile specificate nu exista!');
when others then
raise_application_error(-20023,'Alta Eroare!');
end;
/

select * from table(Numar1 ('Politic','Generalist'));

DROP TYPE numbers;
DROP FUNCTION Numar1;

--9

--Creati o vizualizare care sa contina pentru fiecare redactie, numarul
total de publicatii
--cat si pretul total; dar si pretul minim si maxim.


create or replace view Viz as
select r.nume Nume, sum(nvl(p.pret,0)) valoare , min(p.pret) minim,
max(p.pret) maxim  from
redactii r, publicatii p where r.redid=p.nrred(+) group by r.nume order
by r.nume;

select * from viz;

drop view viz;

--10

-- Utilizand un trigger definiti o constragere asupra pretului unui
publicatii, conform
-- careia acesta nu poate avea valori negativi (sau egale cu zero) si
nici mai mari de 250.

create or replace trigger trig
before insert or update of pret on publicatii
```

```
for each row
begin
if(:new.pret<=0) or (:new.pret>250) then raise_application_error(-20991,'
Nu este permisa introducerea unui pret negativ sau mai mare ca 250!');
end if;
end;
/

update publicatii set
pret=-1 where pubid=3;

insert into publicatii(pubid,nume,pret,nrred) values(15,'Dilema Veche',-
1,2);

update publicatii set
pret=25 where pubid=4;

insert into publicatii(pubid,nume,pret,nrred) values(15,'Dilema
Veche',2,3);

rollback;
drop trigger trig;
```

8 Bibliografie

[1] Materiale de curs SDA, PL/SQL;

[2] Oracle9i Application Developer's Guide - Fundamentals, Release 2 (9.2), 2002.